Design Document

MySQL

We begin with downloading MySQL Community Server version 8.2.0 Innovation. I chose to work with MySQL since it is the database I have worked with before in previous classes and comes with the tool MySQL Workbench which is a useful visual helper tool/IDE that runs your MySQLl server and is easy to work with to visualize queries and errors to help debug as well.

Schema

For the schema, I adhered to the schema provided in the project instructions document and did not add any additional features that I deemed useful. While I did run into problems during my programming where having an additional value in the schema for a specific table may have provided some assistance, I decided to stick with the schema provided as I did not want to run into any trouble regarding my updated schema not necessarily being backwards compatible, nor did I want to have any additional explanations during the evaluation with the TA. I maintained all listed primary and foreign keys and followed the ‘create-company-database.sql’ syntax provided by the professor. For all attributes considered ‘not null’ not explicitly stated in the instructions is assumed to be so. Also, it is assumed any attributes using VARCHAR will have its entries fit into the allotted space. Some additional assumptions for the individual tables include:

- Book:
    - ISBNs used will be ISBN10, while we were given the option to use either ISBN10 or ISBN13, I went with ISBN10 due to project instructions stating: “All book id's are 10-character ISBN numbers.” Furthermore, I did not incorporate and mechanism to convert between the two as it was not required, therefore my project assumes all queries will only work using the provided ISBN10 values.
- Authors:
    - We incorporated MySQL’s ‘auto_increment’ functionality when adding authors from authors.csv so all authors should have their own unique id starting at 1
- Borrower:
    - It is assumed all card IDs follow the format: ‘ID000000’ and any borrowers added will just be one more than the last ID
    - It is assumed all SSNs will follow the format: 123-45-6789 with hyphens included.
    - It is assumed any phone numbers added will follow the format: ‘(123) 456-7890’
- Book_loans
    - We incorporated MySQL’s ‘auto_increment’ functionality when adding loans when checking out so all loans should have their own unique id starting at 1.

Book_parser.py

As stated in the project instructions, part of our system design tasks was to map the data onto our schema and tables. I chose to code a parser in python to make use of the Pandas library which is efficient at dealing with .csv files due to its many useful methods and use of data frames. Furthermore, Pandas is a library I have encountered before but have never had proper practice with, so I wanted to use this project as a way to work with the library and have additional practice in Python programming. In this file, the books.csv and borrowers.csv files are parsed in order to create 4 new .csv files that will be used to fill in the tables in my schema. I went in this direction as I learned of MySQL's 'LOAD DATA LOCAL INFILE' functionality which is able to read in csv files and insert values into tables. The 4 new .csv files are used to insert values for the BOOK, AUTHORS, BOOK_AUTHORS, and BORROWER tables.

Main.java

Now you might be wondering why I did not stick with python for the GUI coding. I initially tried incorporating the tkinter package, however, it did not seem compatible with my laptop (the interface was very glitchy). I switched to Java Swing as it worked much better and after the 'CompanyDBExample.java' example was posted, I was able to follow how the professor connected with MySQL in java. Main.java holds all the main menu GUI and functionalities and when buttons for the different options are pressed, its correlated java file and methods are called upon. I split it up this way in order to keep my code more organized and easy to follow. This is the file that the user runs which should handle all required functionalities. Additionally, for all functionalities opening up menu options, if the menu is already open, the button will not open a new frame to prevent spamming of new menu frames.

SearchBook.java

All necessary search result info is separated by '|'. Users cannot select more than 3 books for checkout since users cannot loan more than 3 books.

Assumptions include:

- If the user presses the search button with no input, all books will be shown in the results
- Partial ISBN search also supports substring matching
- A search containing a combination of ISBN, title, and/or Author(s) must be separated by spaces for search to work as intended
- Titles with spaces will return substring matching for each word, not just the singular book with inputted title
    - While this is not ideal, this was the solution I was able to come up with in order to support substring matching while also being able to search using a combination of values
- Just in case results does not show updated availability, pressing search will act as refresh button

CheckOutBook.java

Assumptions include:

- Inputted Card ID must be valid
- If user tries to check out any number of books that will cause their number of loans to exceed 3, entire transaction is canceled
- If any book in selection is unavailable check out is not allowed
- Check out by ISBN:
    - User must input ISBN exactly correct or check out will fail

Check In Book:

Assumptions include:

- If the user presses the search button with no input, all book loans will be shown in the results
- Partial ISBN or Card ID search also supports substring matching
- A search containing a combination of ISBN, Card ID, and/or Author(s) must be separated by spaces for search to work as intended
- Just in case results does not show updated books currently loaned out, pressing search will act as refresh button

AddBorrower.java

Unique Card ID is created by finding the the last numeric value from the borrower's list and adding 1 (e.g. ID001000 = 1000 so new user will be ID001001)

Assumptions include:

- Phone is optional since not explicitly stated in instructions
- Same information can be entered as long as different SSN

Fines.java

By default, paid fines are not shown and cannot be interacted with as allowed by the instructions. There is no toggle functionality. These were done as this was easier to accomplish and toggle functionality was superfluous.

Assumptions include:

- User themselves must know what books are overdue, display of fines/payment of fines will not indicate which book(s) must be returned before the fine can be paid
- User must put in valid numerical value